

ET4201976030 S

**APPLICATION
FOR
UNITED STATES LETTERS PATENT**

101042078.010802

INTERNATIONAL BUSINESS MACHINES CORPORATION

**A METHOD AND SYSTEM FOR SCHEDULING
EXECUTION OF ACTIVITIES**

Field of the Invention

5 The present invention relates to a method and system for scheduling execution of activities.

Background of the Invention

Scheduling methods are commonly used in several applications, such as for managing a series of jobs in a batch processing. Typically, scheduling involves building a plan consisting of a flow of jobs structured in a specified sequence, which is used to submit and control their execution. The building of the plan requires an accurate prediction of the duration of each job.

20 A solution known in the art is that of measuring an actual duration of each job when it is executed, and using this time as a basis for predicting how long the job should run in the future. For example, an estimated duration of the job is set to an average of the measured durations of completed instances of the job executed previously.

25 A drawback of the solution described above is that it does not provide an accurate prediction of the duration of a new instance of the job. As a consequence, a workload scheduler planning execution of the jobs and controlling their running may operate incorrectly, for

10 008073-15 008073-15

example because a latest start time or a deadline time for the jobs are set to a wrong value.

Moreover, when a job runs significantly longer than its estimated duration the workload scheduler may raise alarms or may invoke a workload manager for giving additional system resources to the job (if classified as critical). However, if the duration of the jobs cannot be predicted accurately, alarms are raised even if no error condition occurred or the system resources are used indiscriminately.

It is an object of the present invention to overcome the above-mentioned drawbacks. In order to achieve this object, a method as set out in the first claim is proposed.

Disclosure of the Invention

Briefly, the present invention provides a method of scheduling execution of a plurality of activities including the steps of estimating a duration of a new instance of each activity to be scheduled as a function of a measured duration of completed instances of the activity executed previously, and adjusting the estimated duration according to the value of at least one attribute of the new instance of the activity.

Moreover, the present invention also provides a computer program for performing the method, a program product storing the program, and a corresponding system.

Brief Description of the Drawings

Further features and the advantages of the solution according to the present invention will be made clear by the following description of a preferred embodiment thereof, given purely by way of a non-restrictive indication, with reference to the attached figures, in which:

Fig.1 is basic block diagram of a data processing system in which the scheduling method of the invention can be used;

Fig.2a shows a partial content of a working memory of the system;

Fig.2b depicts an example of a decision tree used by the scheduling method;

Figg.3a-3b are a flow chart of the scheduling method implemented in the system.

Description of the Preferred Embodiments

With reference in particular to Fig.1, there is shown a data processing system 100, for example a mainframe. The system 100 includes several units, which are connected in parallel to a system bus (SYS_BUS) 105. Particularly, multiple microprocessors (μ P) 110 control operation of the system 100. The microprocessors 110 access a shared working memory 115, generally formed by a DRAM with interleaved modules; a read-only memory (ROM) 120 stores a basic program for starting the system 100.

Various peripheral units are clustered around a local bus (LOC_BUS) 125. More specifically, a bulk memory

consists of a hard-disk 130; the system 100 further includes input units (IN) 135, which consist for example of keyboards and drivers for CD-ROMs 140, and output units (OUT) 145, which consist for example of monitors and printers. A bridge unit (BRIDGE) 150 interfaces the system bus 105 with the local bus 125.

Each microprocessor 105 and the bridge unit 150 can operate as master agents requesting an access to the system bus 105 for transmitting information. The granting of the access with mutual exclusion to the system bus 105 is managed by an arbiter (ARB) 155.

Likewise considerations apply if the system has a different structure (for example with a single bus), if it consists of a mini-computer or a network of workstations, if it includes different units (such as drivers for magnetic tapes), and so on.

Considering now Fig.2, there is shown a partial content of the working memory 115 of the system in operation; the information (programs and data) is typically stored on the hard-disk and loaded (at least partially) into the working memory when the programs are running, together with an operating system and other application programs (not shown in the figure). The programs are initially installed onto the hard disk from CD-ROM.

10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

The working memory 115 includes an input/output interface (I/O) 205, which is used for exchanging information with an operator. The input/output interface 205 communicates with a batch controller (BATCH) 210. The
5 batch controller 210 manages execution of a series of non-interactive jobs (typically during the evening or whenever the system is idle); batch processing is particularly useful for operations that require the system or a peripheral device for an extended period of time.

The batch controller 210 accesses a database 215, which stores a description of an enterprise production workload. The workload database 215 includes a record (accessed through a respective identifier code) for each job of a batch to be executed on the system. The record is composed of a field (JOB) that describes the operations of the job, a planned time of execution (hour, day, week and month), and any dependency from other jobs.
15 In fact, although the system may execute multiple jobs in parallel by a time sharing technique, several jobs often feature sequential constraints. For example, when a resource (such as a file) to be used by a job is being accessed by another job, it is required that the
20 execution of the job should not be started until after the execution of the other job is completed. The record further includes a field (ED) indicating an estimated duration of the job, which is predicted as described in
25 the following.

10
15
20
25
30

30

The descriptions of the jobs to be executed are provided to the batch controller 210, and then transmitted to a plan builder (BUILDER) 220. The builder 220 creates a scheduling plan for controlling a flow of execution of the jobs in a determined sequence for a specified period; the plan is built according to a pre-set scheduling strategy (for example to balance a system load or to optimise a peak performance). The plan is supplied, through the batch controller 210, to a plan executor (EXEC) 225. The plan executor 225 submits the batch in a production environment (RUN) 230, and controls running of the jobs accordingly.

Once a job terminates its execution (because all the operations have been completed or an error has occurred), feedback information are returned to the batch controller 210 from the plan executor 225; the feedback information includes an actual start time of the job, an actual end time of the job, a return code specifying the result of the operations, and the like. The plan executor 225 also measures an actual duration of the job (AD) by subtracting the actual start time from the actual end time. The actual duration AD is fed into an estimation module (EST) 235; the estimation module 235 uses the actual duration AD for predicting how long the job should run in the future. An algorithm is applied to weight the actual duration AD of each run and to create a continually evolving average for the duration of the job; particularly, the actual duration AD is multiplied by a factor with a value decreasing according to the number of

100102078 · 16 1008000

20

25

30

runs of the job, and the result is used for revising the current estimated duration. The revised estimated duration ED is provided to the batch controller 210, and then stored in the corresponding record of the workload database 215.

The plan builder 220 also communicates with a classifier (CLASS) 240. The classifier receives the description of each job to be executed from the plan builder 220, and further accesses a model repository 245. The model repository 245 stores, for each job, a structure defining a decision tree (TREE).

Decision trees are a formalism employed for representing pattern models in classification problems. Classification problems are a branch of a broader scientific discipline known as machine learning, whose scope is the investigation of algorithms that automatically improve their performance at some task through experience (as described for example in "Tom M. Mitchell, Machine Learning, McGraw-Hill, 1997").

Generally, a learning problem is designed by choosing a type of knowledge to be acquired, which is defined by a target function (represented though a chosen formalism), and a corresponding performance measure. The target function is inducted (or learnt) from training experience consisting of known examples of the target function. The learning of the target function involves searching though a space of possible hypotheses to infer

the one that best fits the training examples. One common approach is that of defining the best hypothesis as that which minimises an error between values predicted using the target function and the corresponding actual values.

5

Particularly, classification problems are used for classifying different instances of an event into one of a discrete set of disjoint categories (classes), according to the values of a set of properties (attributes) of each instance of the event. In a classification problem, the target function consists of a classification rule that is used to predict the class of any (hitherto-unseen) new instance of the event from its values of the attributes. The classification rule is typically represented by a decision tree. A (learned) decision tree includes decision nodes specifying a test of some attributes of the event; the branches descending from each node correspond to the possible values of the attribute. Each leaf node specifies the class of the corresponding instances of the event.

10
15
20
25

An instance of the event is classified (i.e., it is assigned to the corresponding class) by sorting it down the tree from the root to the respective leaf node; more specifically, the process involves testing the attribute specified in the root node, then moving down the branch corresponding to the value of the attribute, and repeating the same operations until a leaf node is reached. The performance of the decision tree is measured

as the proportion of the new instances of the event that
are correctly classified.

In the example at issue, each class is associated
with a corresponding adjustment factor (AF). The
adjustment factors (that is the classes to be learnt) are
identified in advance and set to a series of values from
0.1 to 10. The adjustment factors are spaced in a
non-uniform manner; particularly, a difference between
two consecutive adjustment factors from 0.1 to 1 is set
to 0.1, while a difference between two consecutive
adjustment factors from 1 to 10 is set to 0.5. The
decision tree classifies the different instances of the
job according to their planned time of execution;
particularly, the attributes of the job taken into
consideration are an hour of the day, a day of the week,
a day of the month, a day of the year, a week of the
month, a week of the year, and a month of the year.

For each job to be planned, the classifier 240
assigns the new instance of the job to the corresponding
class applying the respective decision tree (according to
the description of the job received from the plan builder
220); the adjustment factor AF of the class assigned to
the new instance of the job is used to revise its
estimated duration ED. Particularly, an adjusted
estimated duration AED is calculated multiplying the
estimated duration ED by the adjustment factor AF, that
is $AED = ED * AF$. The adjusted estimated duration AED is
returned by the classifier 240 to the plan builder 220.

The model repository 245 further includes a log of the feedback information. For each completed instance of the jobs executed previously, the log includes a record (accessed through a respective identifier code) that is composed of a field TIME specifying when the job has been executed, a field AD indicating its actual duration (end time minus start time), a field AED indicating its adjusted estimated duration, and a field AF indicating the (actual) adjustment factor. The adjustment factor AF is calculated dividing the actual duration AD by the estimated duration ED, and rounding off the result to the closest pre-set value of the adjustment factors; for example, if the actual duration is AD=70s and the estimated duration is AF=80s, the adjustment factor will be $AF = AD/ED = 70/80 = 0.875 \approx 0.9$.

The information stored in the model database 245 is updated by a learning module (LEARN) 250. Particularly, the learning module 250 stores the feedback information provided by the execution module 225 and revises the respective decision tree accordingly. The decision tree is built using the feedback information of the completed instances of the job as training examples.

The learning module 250 employs an algorithm operating either in a non-incremental mode or in an incremental mode. When the learning algorithm operates in a non-incremental mode, the decision tree is built from scratch, based on the entire set of available feedback information accumulated in the model repository 245.

The decision tree is preferably built selecting, for each node, the attribute that better separates the completed instances of the job alone; this brings about selection of the simplest decision tree, which is
5 preferred on the ground that it is more likely to capture some meaningful relationship between a job's class and its values of the attributes (according to the so-called Occam's razor). A quantitative measure of the worth of an attribute is for example an information gain, which defines the reduction in entropy caused by partitioning the completed instances of the job according to this attribute, and a split information, which is sensitive to how broadly and uniformly the attribute splits the completed instances of the job. The process is carried out for each node, until either every attribute has been included along the respective path through the tree or the completed instances of the job associated with the node all have the same adjustment factor (that is their entropy is zero). An algorithm of this type is described
10 in "J.R. Quinlan, Induction of Decision Trees", Machine Learning Review, vol.1, pp.81-106, Kluwer Publishers,
15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100 105 110 115 120 125 130 135 140 145 150 155 160 165 170 175 180 185 190 195 200 205 210 215 220 225 230 235 240 245 250 255 260 265 270 275 280 285 290 295 300 305 310 315 320 325 330 335 340 345 350 355 360 365 370 375 380 385 390 395 400 405 410 415 420 425 430 435 440 445 450 455 460 465 470 475 480 485 490 495 500 505 510 515 520 525 530 535 540 545 550 555 560 565 570 575 580 585 590 595 600 605 610 615 620 625 630 635 640 645 650 655 660 665 670 675 680 685 690 695 700 705 710 715 720 725 730 735 740 745 750 755 760 765 770 775 780 785 790 795 800 805 810 815 820 825 830 835 840 845 850 855 860 865 870 875 880 885 890 895 900 905 910 915 920 925 930 935 940 945 950 955 960 965 970 975 980 985 990 995 1000 1005 1010 1015 1020 1025 1030 1035 1040 1045 1050 1055 1060 1065 1070 1075 1080 1085 1090 1095 1100 1105 1110 1115 1120 1125 1130 1135 1140 1145 1150 1155 1160 1165 1170 1175 1180 1185 1190 1195 1200 1205 1210 1215 1220 1225 1230 1235 1240 1245 1250 1255 1260 1265 1270 1275 1280 1285 1290 1295 1300 1305 1310 1315 1320 1325 1330 1335 1340 1345 1350 1355 1360 1365 1370 1375 1380 1385 1390 1395 1400 1405 1410 1415 1420 1425 1430 1435 1440 1445 1450 1455 1460 1465 1470 1475 1480 1485 1490 1495 1500 1505 1510 1515 1520 1525 1530 1535 1540 1545 1550 1555 1560 1565 1570 1575 1580 1585 1590 1595 1600 1605 1610 1615 1620 1625 1630 1635 1640 1645 1650 1655 1660 1665 1670 1675 1680 1685 1690 1695 1700 1705 1710 1715 1720 1725 1730 1735 1740 1745 1750 1755 1760 1765 1770 1775 1780 1785 1790 1795 1800 1805 1810 1815 1820 1825 1830 1835 1840 1845 1850 1855 1860 1865 1870 1875 1880 1885 1890 1895 1900 1905 1910 1915 1920 1925 1930 1935 1940 1945 1950 1955 1960 1965 1970 1975 1980 1985 1990 1995 2000 2005 2010 2015 2020 2025 2030 2035 2040 2045 2050 2055 2060 2065 2070 2075 2080 2085 2090 2095 2100 2105 2110 2115 2120 2125 2130 2135 2140 2145 2150 2155 2160 2165 2170 2175 2180 2185 2190 2195 2200 2205 2210 2215 2220 2225 2230 2235 2240 2245 2250 2255 2260 2265 2270 2275 2280 2285 2290 2295 2300 2305 2310 2315 2320 2325 2330 2335 2340 2345 2350 2355 2360 2365 2370 2375 2380 2385 2390 2395 2400 2405 2410 2415 2420 2425 2430 2435 2440 2445 2450 2455 2460 2465 2470 2475 2480 2485 2490 2495 2500 2505 2510 2515 2520 2525 2530 2535 2540 2545 2550 2555 2560 2565 2570 2575 2580 2585 2590 2595 2600 2605 2610 2615 2620 2625 2630 2635 2640 2645 2650 2655 2660 2665 2670 2675 2680 2685 2690 2695 2700 2705 2710 2715 2720 2725 2730 2735 2740 2745 2750 2755 2760 2765 2770 2775 2780 2785 2790 2795 2800 2805 2810 2815 2820 2825 2830 2835 2840 2845 2850 2855 2860 2865 2870 2875 2880 2885 2890 2895 2900 2905 2910 2915 2920 2925 2930 2935 2940 2945 2950 2955 2960 2965 2970 2975 2980 2985 2990 2995 3000 3005 3010 3015 3020 3025 3030 3035 3040 3045 3050 3055 3060 3065 3070 3075 3080 3085 3090 3095 3100 3105 3110 3115 3120 3125 3130 3135 3140 3145 3150 3155 3160 3165 3170 3175 3180 3185 3190 3195 3200 3205 3210 3215 3220 3225 3230 3235 3240 3245 3250 3255 3260 3265 3270 3275 3280 3285 3290 3295 3300 3305 3310 3315 3320 3325 3330 3335 3340 3345 3350 3355 3360 3365 3370 3375 3380 3385 3390 3395 3400 3405 3410 3415 3420 3425 3430 3435 3440 3445 3450 3455 3460 3465 3470 3475 3480 3485 3490 3495 3500 3505 3510 3515 3520 3525 3530 3535 3540 3545 3550 3555 3560 3565 3570 3575 3580 3585 3590 3595 3600 3605 3610 3615 3620 3625 3630 3635 3640 3645 3650 3655 3660 3665 3670 3675 3680 3685 3690 3695 3700 3705 3710 3715 3720 3725 3730 3735 3740 3745 3750 3755 3760 3765 3770 3775 3780 3785 3790 3795 3800 3805 3810 3815 3820 3825 3830 3835 3840 3845 3850 3855 3860 3865 3870 3875 3880 3885 3890 3895 3900 3905 3910 3915 3920 3925 3930 3935 3940 3945 3950 3955 3960 3965 3970 3975 3980 3985 3990 3995 4000 4005 4010 4015 4020 4025 4030 4035 4040 4045 4050 4055 4060 4065 4070 4075 4080 4085 4090 4095 4100 4105 4110 4115 4120 4125 4130 4135 4140 4145 4150 4155 4160 4165 4170 4175 4180 4185 4190 4195 4200 4205 4210 4215 4220 4225 4230 4235 4240 4245 4250 4255 4260 4265 4270 4275 4280 4285 4290 4295 4300 4305 4310 4315 4320 4325 4330 4335 4340 4345 4350 4355 4360 4365 4370 4375 4380 4385 4390 4395 4400 4405 4410 4415 4420 4425 4430 4435 4440 4445 4450 4455 4460 4465 4470 4475 4480 4485 4490 4495 4500 4505 4510 4515 4520 4525 4530 4535 4540 4545 4550 4555 4560 4565 4570 4575 4580 4585 4590 4595 4600 4605 4610 4615 4620 4625 4630 4635 4640 4645 4650 4655 4660 4665 4670 4675 4680 4685 4690 4695 4700 4705 4710 4715 4720 4725 4730 4735 4740 4745 4750 4755 4760 4765 4770 4775 4780 4785 4790 4795 4800 4805 4810 4815 4820 4825 4830 4835 4840 4845 4850 4855 4860 4865 4870 4875 4880 4885 4890 4895 4900 4905 4910 4915 4920 4925 4930 4935 4940 4945 4950 4955 4960 4965 4970 4975 4980 4985 4990 4995 5000 5005 5010 5015 5020 5025 5030 5035 5040 5045 5050 5055 5060 5065 5070 5075 5080 5085 5090 5095 5100 5105 5110 5115 5120 5125 5130 5135 5140 5145 5150 5155 5160 5165 5170 5175 5180 5185 5190 5195 5200 5205 5210 5215 5220 5225 5230 5235 5240 5245 5250 5255 5260 5265 5270 5275 5280 5285 5290 5295 5300 5305 5310 5315 5320 5325 5330 5335 5340 5345 5350 5355 5360 5365 5370 5375 5380 5385 5390 5395 5400 5405 5410 5415 5420 5425 5430 5435 5440 5445 5450 5455 5460 5465 5470 5475 5480 5485 5490 5495 5500 5505 5510 5515 5520 5525 5530 5535 5540 5545 5550 5555 5560 5565 5570 5575 5580 5585 5590 5595 5600 5605 5610 5615 5620 5625 5630 5635 5640 5645 5650 5655 5660 5665 5670 5675 5680 5685 5690 5695 5700 5705 5710 5715 5720 5725 5730 5735 5740 5745 5750 5755 5760 5765 5770 5775 5780 5785 5790 5795 5800 5805 5810 5815 5820 5825 5830 5835 5840 5845 5850 5855 5860 5865 5870 5875 5880 5885 5890 5895 5900 5905 5910 5915 5920 5925 5930 5935 5940 5945 5950 5955 5960 5965 5970 5975 5980 5985 5990 5995 6000 6005 6010 6015 6020 6025 6030 6035 6040 6045 6050 6055 6060 6065 6070 6075 6080 6085 6090 6095 6100 6105 6110 6115 6120 6125 6130 6135 6140 6145 6150 6155 6160 6165 6170 6175 6180 6185 6190 6195 6200 6205 6210 6215 6220 6225 6230 6235 6240 6245 6250 6255 6260 6265 6270 6275 6280 6285 6290 6295 6300 6305 6310 6315 6320 6325 6330 6335 6340 6345 6350 6355 6360 6365 6370 6375 6380 6385 6390 6395 6400 6405 6410 6415 6420 6425 6430 6435 6440 6445 6450 6455 6460 6465 6470 6475 6480 6485 6490 6495 6500 6505 6510 6515 6520 6525 6530 6535 6540 6545 6550 6555 6560 6565 6570 6575 6580 6585 6590 6595 6600 6605 6610 6615 6620 6625 6630 6635 6640 6645 6650 6655 6660 6665 6670 6675 6680 6685 6690 6695 6700 6705 6710 6715 6720 6725 6730 6735 6740 6745 6750 6755 6760 6765 6770 6775 6780 6785 6790 6795 6800 6805 6810 6815 6820 6825 6830 6835 6840 6845 6850 6855 6860 6865 6870 6875 6880 6885 6890 6895 6900 6905 6910 6915 6920 6925 6930 6935 6940 6945 6950 6955 6960 6965 6970 6975 6980 6985 6990 6995 7000 7005 7010 7015 7020 7025 7030 7035 7040 7045 7050 7055 7060 7065 7070 7075 7080 7085 7090 7095 7100 7105 7110 7115 7120 7125 7130 7135 7140 7145 7150 7155 7160 7165 7170 7175 7180 7185 7190 7195 7200 7205 7210 7215 7220 7225 7230 7235 7240 7245 7250 7255 7260 7265 7270 7275 7280 7285 7290 7295 7300 7305 7310 7315 7320 7325 7330 7335 7340 7345 7350 7355 7360 7365 7370 7375 7380 7385 7390 7395 7400 7405 7410 7415 7420 7425 7430 7435 7440 7445 7450 7455 7460 7465 7470 7475 7480 7485 7490 7495 7500 7505 7510 7515 7520 7525 7530 7535 7540 7545 7550 7555 7560 7565 7570 7575 7580 7585 7590 7595 7600 7605 7610 7615 7620 7625 7630 7635 7640 7645 7650 7655 7660 7665 7670 7675 7680 7685 7690 7695 7700 7705 7710 7715 7720 7725 7730 7735 7740 7745 7750 7755 7760 7765 7770 7775 7780 7785 7790 7795 7800 7805 7810 7815 7820 7825 7830 7835 7840 7845 7850 7855 7860 7865 7870 7875 7880 7885 7890 7895 7900 7905 7910 7915 7920 7925 7930 7935 7940 7945 7950 7955 7960 7965 7970 7975 7980 7985 7990 7995 8000 8005 8010 8015 8020 8025 8030 8035 8040 8045 8050 8055 8060 8065 8070 8075 8080 8085 8090 8095 8100 8105 8110 8115 8120 8125 8130 8135 8140 8145 8150 8155 8160 8165 8170 8175 8180 8185 8190 8195 8200 8205 8210 8215 8220 8225 8230 8235 8240 8245 8250 8255 8260 8265 8270 8275 8280 8285 8290 8295 8300 8305 8310 8315 8320 8325 8330 8335 8340 8345 8350 8355 8360 8365 8370 8375 8380 8385 8390 8395 8400 8405 8410 8415 8420 8425 8430 8435 8440 8445 8450 8455 8460 8465 8470 8475 8480 8485 8490 8495 8500 8505 8510 8515 8520 8525 8530 8535 8540 8545 8550 8555 8560 8565 8570 8575 8580 8585 8590 8595 8600 8605 8610 8615 8620 8625 8630 8635 8640 8645 8650 8655 8660 8665 8670 8675 8680 8685 8690 8695 8700 8705 8710 8715 8720 8725 8730 8735 8740 8745 8750 8755 8760 8765 8770 8775 8780 8785 8790 8795 8800 8805 8810 8815 8820 8825 8830 8835 8840 8845 8850 8855 8860 8865 8870 8875 8880 8885 8890 8895 8900 8905 8910 8915 8920 8925 8930 8935 8940 8945 8950 8955 8960 8965 8970 8975 8980 8985 8990 8995 9000 9005 9010 9015 9020 9025 9030 9035 9040 9045 9050 9055 9060 9065 9070 9075 9080 9085 9090 9095 9100 9105 9110 9115 9120 9125 9130 9135 9140 9145 9150 9155 9160 9165 9170 9175 9180 9185 9190 9195 9200 9205 9210 9215 9220 9225 9230 9235 9240 9245 9250 9255 9260 9265 9270 9275 9280 9285 9290 9295 9300 9305 9310 9315 9320 9325 9330 9335 9340 9345 9350 9355 9360 9365 9370 9375 9380 9385 9390 9395 9400 9405 9410 9415 9420 9425 9430 9435 9440 9445 9450 9455 9460 9465 9470 9475 9480 9485 9490 9495 9500 9505 9510 9515 9520 9525 9530 9535 9540 9545 9550 9555 9560 9565 9570 9575 9580 9585 9590 9595 9600 9605 9610 9615 9620 9625 9630 9635 9640 9645 9650 9655 9660 9665 9670 9675 9680 9685 9690 9695 9700 9705 9710 9715 9720 9725 9730 9735 9740 9745 9750 9755 9760 9765 9770 9775 9780 9785 9790 9795 9800 9805 9810 9815 9820 9825 9830 9835 9840 9845 9850 9855 9860 9865 9870 9875 9880 9885 9890 9895 9900 9905 9910 9915 9920 9925 9930 9935 9940 9945 9950 9955 9960 9965 9970 9975 9980 9985 9990 9995 9999 10000 10005 10010 10015 10020 10025 10030 10035 10040 10045 10050 10055 10060 10065 10070 10075 10080 10085 10090 10095 10100 10105 10110 10115 10120 10125 10130 10135 10140 10145 10150 10155 10160 10165 10170 10175 10180 10185 10190 10195 10200 10205 10210 10215 10220 10225 10230 10235 10240 10245 10250 10255 10260 10265 10270 10275 10280 10285 10290 10295 10300 10305 10310 10315 10320 10325 10330 10335 10340 10345 10350 10355 10360 10365 10370 10375 10380 10385 10390 10395 10400 10405 10410 10415 10420 10425 10430 10435 10440 10445 10450 10455 10460 10465 10470 10475 10480 10485 10490 10495 10500 10505 10510 10515 10520 10525 10530 10535 10540 10545 10550 10555 10560 10565 10570 10575 10580 10585 10590 10595 10600 10605 10610 10615 10620 10625 10630 10635 10640 10645 10650 10655 10660 10665 10670 10675 10680 10685 10690 10695 10700 10705 10710 10715 10720 10725 10730 10735 10740 10745 10750 10755 10760 10765 10770 10775 10780 10785 10790 10795 10800 10805 10810 10815 10820 10825 10830 10835 10840 10845 10850 10855 10860 10865 10870 10875 10880 10885 10890 10895 10900 10905 10910 10915 10920 10925 10930 10935 10940 10945 10950 10955 10960 10965 10970 10975 10980 10985 10990 10995 10999 11000 11005 11010 11015 11020 11025 11030 11035 11040 11045 11050 11055 11060 11065 11070 11075 11080 11085 11090 11095 11100 11105 11110 11115 11120 11125 11130 11135 11140 11145 11150 11155 11160 11165 11170 11175 11180 11185 11190 11195 11200 11205 11210 11215 11220 11225 11230 11235 11240 11245 11250 11255 11260 11265 11270 11275 11280 11285 11290 11295 11300 11305 11310 11315 11320 11325 11330 11335 11340 11345 11350 11355 11360 11365 11370 11375 11380 11385 11390 11395 11400 11405 11410 11415 11420 11425 11430 11435 11440 11445 11450 11455 11460 11465 11470 11475 11480 11485 11490 11495 11500 11505 11510 11515 11520 11525 11530 11535 11540 11545 11550 11555 11560 11565 11570 11575 11580 11585 11590 11595 11600 11605 11610 11615 11620 11625 11630 11635 11640 11645 11650 11655 11660 11665 11670 11675 11680 11685 11690 11695 11700 11705 11710 11715 11720 11725 11730 11735 11740 11745 11750 11755 11760 11765 11770 11775 11780 11785 11790 11795 11800 11805 11810 11815 11820 11825 11830 11835 11840 11845 11850 11855 11860 11865 11870 11875 11880 11885 1

information is saved at the leaf node. Whenever a further instance of the job is to be incorporated, the decision tree is followed according to the attributes of the further instance of the job, as far as a leaf node is reached. If the further instance of the job belongs to the same class as the leaf node, it is simply added to the set of instances of the job saved at the leaf node. Conversely, the algorithm attempts to turn the leaf node into a decision node; the instances of the job saved at the leaf node just converted to a decision node are then incorporated recursively down their proper branch according to the new test. Preferably, the tree is also restructured as necessary so that the best test attribute is at each decision node. It should be noted that, although the feedback information of the various instances of the job is retained within the decision tree structure, it is only used for restructuring purposes, and is not reprocessed each time a new instance of the job is observed. An algorithm of this type is described in "Paul E. Utgoff, Incremental Induction of Decision Trees, Machine Learning Review, vol.4, pp.161-186, Kluwer Publisher, 1989" and "Paul E. Utgoff, Decision Tree Induction Based on Efficient Tree Restructuring, Machine Learning Review, vol.29, pp.5-44, 1997".

An example of the decision tree used by the scheduling method of the invention is shown in Fig.2b. The decision tree includes a root node based on the day of week in which the job runs. The job duration is constant on Monday through Thursday (irrespective of the

value of the other attributes). On Saturdays and Sundays the job duration is less than the running average (adjustment factor AF=0.4), except on the last week of the month (WK4) when the job is very long-running (adjustment factor AF=10). When the job runs on Fridays, its duration varies both on the month and on the week of the month.

If we apply this decision tree to a new instance of the job planned to be executed at the following time:

```
Hour of day = 8  
Day of week = Friday  
Day of month = 9  
Month of year = January  
Week of month = WK2
```

the classifier goes through the various nodes of the decision tree and finds that the class of the new instance of the job has an adjustment factor AF=3.5. Its estimated duration (for example ED=120s) is then multiplied by the adjustment factor to provide the more accurate adjusted estimated duration AED = ED*AF = $120 * 3.5 = 420s$.

The performance of the algorithm is measured by a prediction error ERR%, which is defined as the percentage of an absolute value of the difference between the adjusted estimated duration of each instance of the job

and its actual duration with respect to the actual duration ($ERR\% = |AED-AD| * 100/AD$).

Likewise considerations apply if a whole scheduling program (consisting of the plan builder, the estimation module, the classifier and the learning module) and the data are structured in a different manner, for example if the workload database and the model repository consist of a single inventory or they are replaced by equivalent memory structures, if different functions are provided, if the estimated duration is predicted in a different manner (for example accumulating the actual duration of each completed instance of the job and calculating their actual running average, or more generally as a function of the measured duration of each completed instance of the job), and so on.

With reference now to Figg.3a-3b, the scheduling program (when running on the system) performs a method 300 that starts at block 305. The method then passes to block 310, wherein the operator requires building of a scheduling plan for a selected batch of jobs. Passing to block 315, the descriptions of the jobs to be executed are retrieved from the workload database by the batch controller, and then transmitted to the plan builder.

The plan builder passes the description of a first job of the batch to the classifier at block 325. The classifier assigns the new instance of the job to the corresponding class applying the respective decision tree

10
05/2078
15
208047

20

25

30

stored in the model repository (according to its planned time of execution). The adjustment estimated duration AED for the new instance of the job is calculated at block 330 multiplying the respective estimated duration ED by the adjustment factor AF of the class assigned to the new instance of the job; the adjusted estimated duration AED is then returned to the plan builder. The method checks at block 335 whether a last job of the batch has been processed. If not, the method returns to block 315 for repeating the operations described above. Conversely, the method descends into block 340.

Considering now block 340, the builder creates the scheduling plan for controlling execution of the jobs according to their adjusted estimated duration; the plan is returned to the batch controller and then passed to the plan executor. The executor submits the plan at block 345, and controls running of each job according to its planned time of execution. The method then enters an idle loop at block 350, waiting for termination of a job execution. If the return code of the job specifies that an error has occurred, the method descends into block 355 (described in the following). Conversely, if the job has been executed correctly the method continues to block 360, wherein the feedback information of the job is returned to the plan executor.

The method then forks into two branches that are executed in parallel. A first branch consists of blocks

10
10002007232000
15
15002007232000
20
20002007232000
25
25002007232000

365-367, and a second branch consists of blocks 370-385; the two branches joint at block 355.

Considering now block 365, the plan executor calculates the actual duration AD of the job (actual end time minus actual start time), which is supplied to the estimation module. The estimation module revises the estimated duration accordingly at block 367; the revised estimated duration ED is then provided to the batch controller and stored in the workload database.

At the same time, at block 370, the feedback information is supplied to the learning module and logged into the model repository. The method checks at block 375 whether the prediction error for the job reaches a threshold value, for example 10%. If the prediction error is lower than the threshold value, the method continues to block 355 directly. Conversely, if the prediction error exceeds the threshold value (or it is equal thereto) the method checks at block 380 whether the operator has selected an incremental mode of operation. If so, the method continues to block 382 wherein the decision tree is revised in the incremental mode according to the feedback information of the job that has been just executed; conversely, the method proceeds to block 385, wherein the decision tree is rebuilt using the feedback information of all the completed instances of the job logged in the model repository. In both cases, the method then descends into block 355.

Considering now block 355, the batch controller checks whether a last job of the plan has been executed. If not, the method returns to block 350, waiting for termination of a further job. On the contrary, the method verifies at block 390 whether the scheduling program has been stopped. If not, the method return to block 310 for repeating the steps described above. Conversely, the method ends at the final block 395.

Likewise considerations apply if the program performs an equivalent method, for example with error routines, escape functions for stopping execution of the jobs, monitoring routines for tracking the jobs, and the like.

More generally, the present invention provides a method of scheduling execution of a plurality of activities. The method includes the steps of estimating a duration of a new instance of each activity to be scheduled as a function of a measured duration of completed instances of the activity executed previously, and adjusting the estimated duration according to the value of at least one attribute of the new instance of the activity.

The devised solution results in a very accurate prediction of the duration of a new instance of each job. This improves operation of the workload scheduler, which generates more reliable scheduling plans. As a consequence, any alarm raised by the workload scheduler

always indicates an error condition, due to a job running significantly longer than its estimated duration. The system resources may also be used correctly by the workload manager for critical jobs that run longer than their estimated duration.

In fact, the inventors have discovered that the various instances of a same job are not always equal; conversely, the duration of the different instances may vary widely due to consistently repeatable factors. The method of the invention recognises the combination of circumstances that can alter the duration of each instance of the job; the estimated duration of a new instance the job is then adjusted taking into account some key information thereof.

The preferred embodiment of the invention described above offers further advantages. For example, the method induces a pattern model that partitions the completed instances of each job into classes defining corresponding adjustment factors; a new instance of the job is assigned to the corresponding class applying the pattern model, and the estimated duration is corrected according to the adjustment factor of the assigned class. In this way, historical information is used to identify repeatable factors that alter the job duration; moreover, the performance of the algorithm may be improved through experience. Preferably, the pattern model is represented by a decision tree; this formalism is particular simple, and at the same time very accurate.

Similar considerations apply if the decision tree has a different structure (with one or more decision nodes and a plurality of leaf nodes), if the pattern model is inducted in a different manner, for example using training examples prepared by an expert, if a different procedure is used for adjusting the estimated duration, and the like. However, the method of the invention leads itself to be carried out even representing the pattern model with different formalisms (for example neural networks, polynomial or linear functions, Bayesian networks), if the adjusted estimated duration is not determined using a classification rule (for example it is calculated using a different target function of a generic learning algorithm), or even if the estimated duration is simply adjusted according to the value of one or more attributes of the new instance of the job (for example using a pre-set adjustment table).

The decision tree is revised or rebuilt when the prediction error reaches a threshold value; in this way, the accuracy of the method is always kept at a desired level. Advantageously, the decision tree is learnt using an algorithm operating in an incremental mode. The incremental mode is preferred for a serial learning task (based on a stream of observed examples), since it is more efficient to refine the decision tree as new training examples become available than to generate a new decision tree each time a new training example is observed.

10
15
20
25
30

Likewise considerations apply if the learning algorithm operates in a different manner (for example without storing the feedback information at each node for restructuring the decision tree, or incorporating a newly observed instance of the job only if the existing decision tree would misclassify it), if the decision tree is built with other techniques (for example pruning the decision tree for avoiding over-fitting due to noise or coincidental regularities in the training examples), if the threshold value is different, and so on.

Alternatively, the decision tree is rebuilt periodically (for example every week), or the learning algorithm operates only in the incremental mode or in the non-incremental mode.

Advantageously, the adjustment factors are set to a series of values from 0.1 to 10. Experimental results have shown these limit values provide a good accuracy of the scheduling method. In fact, extremely short-lived jobs (performing simple operations, such as copying a file, or consisting of dummy jobs used to synchronise other jobs) may vary in duration from one instance to the other, even by large factors; however, the actual duration is always a very small unit of time, so that the variance is relatively uninteresting for planning purposes. Similarly, long running jobs (for example daemon processes) do not have successor dependencies (i.e., no jobs are triggered by their completion); therefore, the estimated duration for these jobs is again relatively uninteresting when building the plan.

Moreover, the adjustment factors are spaced in a non-uniform manner, with the difference between two consecutive adjustment factors that increases with their value. In this way, the adjustment of the estimated duration has a resolution that is high when the duration of the new instance of the job is close to the estimated one; on the other hand, the resolution decreases as far as the duration of the new instance of the job varies by a large factor with respect to the estimated one, since a very accurate prediction is less interesting in this case.

Similar considerations apply if the adjustment factors are spaced in a different manner, for example with the difference between two consecutive adjustment factors that is proportional to their average. However, the solution of the invention leads itself to be carried out even with the adjustment factors uniformly spaced, with the adjustment factors set to different values, or with the adjustment factors, their spacing and/or the limit values that may be customised by the operator to allow tuning to a particular purpose.

Preferably, the decision tree is built using the actual time of execution of the completed instances of the job, while each new instance of the job is classified taking into consideration its planned time of execution. In fact, these attributes are particularly suitable to recognise the combinations of circumstances that in the past have altered the execution of the job; therefore,

they provide a quite accurate prediction of the job duration. For instance, a daily job that processes bank transactions to update a central database may run longer towards the end of the month, when more people use banks to pay bills, deposit or withdraw salary, pay mortgages, and so on. Likewise, it is probable that this job will take less time at weekends when branch offices are closed than it would on weekdays when there are more transactions to process.

Similar considerations apply if the planned/actual time of execution is defined by different parameters, such as only the day of the week and the day of the month. However, the method according to the present invention leads itself to be implemented even taking into consideration one or more different attributes of the job, such as the number of jobs concurrently executed on the system or a microprocessor load at its planned/actual time of execution.

In the preferred embodiment of the invention, the method is used for scheduling execution of jobs in a batch processing. However, different applications of the devised solution are not excluded, such as for controlling system resources in a distributed computer network.

Advantageously, the solution according to the present invention is implemented with a computer program (software), which is provided on CD-ROM.

Alternatively, the program is provided on floppy-disk or tape, is pre-loaded onto the hard-disk, or is stored on any other computer readable medium, is sent to the system through a network (typically the INTERNET), is broadcast, or more generally is provided in any other form directly loadable into a working memory of the computer. However, the method according to the present invention leads itself to be carried out even with a hardware structure, for example integrated in a chip of semiconductor material.

Naturally, in order to satisfy local and specific requirements, a person skilled in the art may apply to the solution described above many modifications and alterations all of which, however, are included within the scope of protection of the invention as defined by the following claims.

10 012078
15 14380E